Model Predictive Control - EL2700

Assignment 4 : Model Predictive Control

2020

Automatic Control
School of Electrical Engineering and Computer Science
Kungliga Tekniska Högskolan

Introduction

In this assignment, we will implement a linear MPC to achieve swing-up and balancing control of an inverted pendulum on a cart. In Part 1, we start by considering an inverted pendulum with simplified cart dynamics. We will compute backwards reachable sets to visualize the feasible initial states of the problem, and design an MPC to stabilize the inverted pendulum through cart acceleration. In Part 2, we will include the dynamics of the cart and implement an error free reference tracking MPC for moving the cart from one position to the other while maintaining the pendulum upright.

Code Organization This time around, we will use Matlab and the MPT3 Toolbox for Part I of this assignment, and Python with CasADi for Parts II and III.

Matlab Code In Matlab we will use the MPT3 Toolbox, which will be helpful for set calculations. To download and install MPT3 Toolbox[1], run `install_mpt3.m`[2] script in Matlab, which can be downloaded from the MPT3 website under "Automatic installation". If you have troubles with this step, you might need to install the Gurobi[3] academic license solver - it can be requested on the Gurobi website.

Python Code Similarly to the previous assignments, the Python code entry-point is `task4.py`. Please run the script `install_deps.py` with Python 3 to install any missing dependencies.

Part I - Sets and Constraints

MPC formulation We will now design a linear MPC to stabilize the inverted pendulum through cart acceleration. The discrete-time linearized dynamics of the inverted pendulum subjected to state and control constraints can be represented by

$$x_{t+1} = Ax_t + Bu_t, \quad y_t = Cx_t \tag{1}$$
$$u_t \in \mathcal{U}, \quad x_t \in \mathcal{X}, \quad t \geq 0 \tag{2}$$

with $x_t = \begin{bmatrix}\theta & \dot{\theta}\end{bmatrix}^T \in \mathbb{R}^2$, $u_T \in \mathbb{R}$, and $y_t = \theta \in \mathbb{R}$. The sets $\mathcal{U}, \mathcal{X}$ are polyhedrons described by the following linear inequalities

$$\mathcal{U} = \{u : H_u u \leq h_u\}, \quad \mathcal{X} = \{x : H_x x \leq h_x\} \tag{3}$$

In this design task, we will consider the following state and control constraints

$$x_{min} \leq x_t \leq x_{max}, \quad u_{min} \leq u_t \leq u_{max} \tag{4}$$

with $x_{max} = -x_{min} = \begin{bmatrix}\pi/2 & \pi/2\end{bmatrix}^T$ and $u_{max} = -u_{min} = 5$. We will now perform finite-horizon optimization in a receding-horizon fashion and solve the following optimization problem

$$\min_{u_{t+\cdot|t}} \sum_{k=0}^{N-1} x_{t+k|t}^T Q x_{t+k|t} + u_{t+k|t}^T R u_{t+k|t} + x_{t+N|t}^T P x_{t+N|t} \tag{5}$$

$$\begin{aligned}
\text{subject to: } & x_{t+k+1|t} = Ax_{t+k|t} + Bu_{t+k|t} && (6)\\
& x_{t+k|t} \in \mathcal{X} && (7)\\
& u_{t+k|t} \in \mathcal{U} && (8)\\
& x_{t+N|t} \in \mathcal{X}_f && (9)\\
& x_{t|t} = x_t && (10)
\end{aligned}$$

with $Q_f \succeq 0$, and $\mathcal{X}_f$ are the terminal cost and terminal constraint set respectively. If the terminal constraint set $\mathcal{X}_f$ is control invariant, then the MPC problem (5) is recursively feasible. In this design task, we will consider the following two control invariant sets as the terminal constraint set:

- the zero terminal constraint set $\mathcal{X}_f = \{0\}$;

[1] MPT3 Toolbox: `https://www.mpt3.org/Main/HomePage`
[2] For more information, please visit `https://www.mpt3.org/Main/Installation`
[3] Website: https://www.gurobi.com/

- the invariant set for the closed-loop dynamics under the infinite-horizon LQR control $\mathcal{X}_f^{LQR}$.

For the MPC problem (2) to be feasible, the controller must be able to steer the system state x_t to the terminal set $\mathcal{X}_f$ in N steps, while satisfying the state and control constraints. To this end, we will first compute the N-step controllable set $\mathcal{K}_N(\mathcal{S})$. We provide you the Matlab script `compute_N_step_controllable_set.m` to compute $\mathcal{K}_N(\mathcal{S})$. Your tasks are as follows:

1. **Q1: Influence of terminal sets**
   Comment on how the size of the terminal set influences the N-step controllable set.

2. For the terminal sets $\mathcal{X}_f = \{0\}$ and $\mathcal{X}_f^{LQR}$, respectively:
   Q2: Study the influence of the control horizon Compute the control invariant set $\mathcal{K}_N(\mathcal{X}_N)$ for varying control horizons $N = 5, 10$, and 20. Keep the state and control constraints as given in (1). Comment on the influence of N on $\mathcal{K}_N(\mathcal{X}_N)$
   Q3: Influence of control constraints Compute the control invariant set $\mathcal{K}_N(\mathcal{X}_N)$ for varying control constraints. Keep the state constraints as given in (4) and use $N = 5$. Comment on the influence of the control constraint on $\mathcal{K}_N(\mathcal{X}_N)$.

Part II - MPC for Inverted Pendulum Stabilization

You will now simulate an inverted pendulum with MPC control by running the Part II of `task4.py`. The initial state for the simulation is set to $x_0 = \begin{bmatrix}-\pi/4 & 0\end{bmatrix}$.

Q4: Examine the MPC controller implementation under the file `mpc.py`. Explain using your own words (you can and should take code snippets to help with the explanation):

1. How the state constraints are set;
2. How the object function is formulated;
3. How the terminal constraint is set;
4. What the variable `params` holds;
5. Why we only select the first element of the control prediction horizon in the `mpc_controller` method.

Q5: At this point, set the control bounds to $u_{max} = -u_{min} = 5$. What happens to the system? Provide some intuition.

Q6: Set as terminal set the sets: (i) $\mathcal{X}_f = \{0\}$, and (ii) $\mathcal{X}_f^{LQR}$. Do you notice performance changes? And with smaller, horizon lengths?

Part III - MPC for Pendulum-Cart System

We will now design a linear MPC for an inverted pendulum including the dynamics of the cart. The objective is to move the cart from one point to the other while keeping the pendulum upright. The controller to be designed should meet the following three performance criteria:

1. The magnitude of the control input should be limited to $\|u_t\| \leq 5$
2. The pendulum should be kept within $\pm 10^\circ$ around the vertical position while the cart should avoid overshoot.

To this end, we will implement reference tracking MPC. We will reuse the discrete-time linear model of the cart pendulum system derived in the first design project

$$x_{t+1} = Ax_t + Bu_t + B_w w_t, \quad y_t = Cx_t \tag{11}$$
$$u_t \in \mathcal{U}, \quad x_t \in \mathcal{X}, \quad t \geq 0 \tag{12}$$

to simulate our system, where $x_t \in \mathbb{R}^4, u_t \in \mathbb{R}, w_t \in \mathbb{R}$ and $y_t \in \mathbb{R}$ are the system state, control input, disturbance input, and output respectively. First, we will start with $w_t = 0$ using the dynamics `pendulum.discrete_time_dynamics`, and after we will simulate with $w_t = 0.05$ using `pendulum.pendulum_linear_dynamics_with_disturbance`.

Q7: Set the reference to $10m$ for the cart position. Comment on the performance of the MPC controller.

Q8: Now set a disturbance in the system equal to the one in the LQR for **Q2** in Assignment 3. Compare the performance of both controllers. Suggest a possible way to fix the steady-state error for the MPC formulation.

As always, refer to the Slack workspace `el2700workspace.slack.com` for questions.

Good Luck!